**预定义过滤**

**ISoftDelete**

软删除

public class Person : Entity, **ISoftDelete**

{

public virtual string Name { get; set; }

**public virtual bool IsDeleted { get; set; }**

}

**IMustHaveTenant**

如果你正在创建多租户应用并存储所有租户数据在一个数据库里，你明确地不想一个租户的数据意外地被另一个租户看到，这种情况下你可用IMustHaveTenant。例如：

public class Product : Entity, **IMustHaveTenant**

{

**public int TenantId { get; set; }**

public string Name { get; set; }

}

IMustHaveTenant定义了TenantId，区别不同的租户实体。ABP默认情况下使用[IAbpSeesion](http://www.cnblogs.com/kid1412/p/5986611.html" \t "http://www.cnblogs.com/kid1412/p/_blank)获取当前TenantId，并自动为当前租户过滤查询。

**IMayHaveTenant**

如果一个实体类被租户和宿主共享（也就是说一个实体对象可被租户或宿主拥有），你可以使用IMayHaveTenant过滤。IMayHaveTenant接口定义了TenantId，但它是可空的。

public class Role : Entity, **IMayHaveTenant**

{

**public int? TenantId { get; set; }**

public string RoleName { get; set; }

}

一个null值表示这是个宿主实体，一个非null值表示这个实体被Id为TenantId的租户拥有。

**禁用过滤**

调用DisableFilter方法可以禁用每个工作单元的一个过滤，如下：

var people1 = \_personRepository.GetAllList();

using (**\_unitOfWorkManager.Current.DisableFilter(AbpDataFilters.SoftDelete)**)

{

var people2 = \_personRepository.GetAllList();

}

var people3 = \_personRepository.GetAllList();

DisableFilter接受一个或多个过滤名称组成的字符串，AbpDataFilters.SoftDelete是个字符串常量，它代表ABP的软删除过滤。

**设置过滤参数**

另一个例子：为IMayHaveTenant过滤，设置租户Id：

CurrentUnitOfWork.SetFilterParameter(AbpDataFilters.MayHaveTenant, AbpDataFilters.Parameters.TenantId, 42);

**SetTenantId 方法**

虽然你可以使用SetFilterParameter方法，为MayHaveTenant和MusthaveTenant修改过滤值，但修改租户过滤有一个更好的方式：SetTenantId()。SetTenantId为这两个过滤修改参数值，并且单数据库或每个租户一个数据库都有效。所以，总是推荐用SetTenantId修改租户过滤的参数值。查看[多租户文档](http://www.cnblogs.com/kid1412/p/5978314.html" \t "http://www.cnblogs.com/kid1412/p/_blank)获取更多信息。